Hacking y Pentesting con Python

Módulo 1: OSINT con Python y recolección de información activa.

eip ESCUELA INTERNACIONAL POSGRADOS

ÍNDICE

Módulo 1: OSINT con Python y recolección de información activa.

PRESENTACIÓN Y OBJETIVOS

En este primer módulo se enseñan las técnicas básicas de recolección de información en fuentes abiertas, también conocidas como OSINT así como las librerías y servicios disponibles en Python para automatizar dichos procesos. A continuación, se explicará cómo llevar a cabo el reconocimiento activo de objetivos mediante las técnicas de escaneo y enumeración que son habituales en procedimientos de pentesting.

Objetivos

- Conocer las diferentes fuentes de información disponibles y automatizar su acceso mediante Python.
- Aprender a utilizar las librerías más comunes en el Python para recolección de información pasiva y activa.
- Aprender a ejecutar procedimientos para la enumeración activa de objetivos.
- Definir las bases para la creación de herramientas en Python orientadas al pentesting y Hacking.

Recolección de información pasiva

1. Introducción

En el mundo del pentesting y el hacking una de las mejores armas que tienen los profesionales de la seguridad informática es precisamente la información que se encuentra disponible en múltiples diversas fuentes. Sin embargo, el reto es precisamente determinar qué información es relevante y cuál representa simplemente datos sin trascendencia. Desde la perspectiva del atacante, es necesario contar con una buena capacidad para extraer y procesar información sobre el sistema objetivo y saber identificar cualquier fallo o fuga de información sensible.

Cuando se habla de la anatomía de un ataque y de seguridad ofensiva, la recopilación y análisis de información es un proceso transversal que se debe implementar en todas las etapas, desde el reconocimiento inicial del objetivo hasta las etapas finales que consisten en la limpieza de rastros y creación de puertas traseras para intentar garantizar accesos futuros. En Python existen una gran variedad de herramientas y librerías que permiten extraer información para realizar actividades de reconocimiento inicial. En los siguientes apartados se detallarán algunas de las más comunes y utilizadas para la recolección de información tanto de forma pasiva como activa.

2. TÉCNICAS OSINT APLICADAS A PYTHON

En primer lugar, uno de los recursos más interesantes y conocidos en el mundo OSINT es precisamente el OSINT Frmework (https://osintframework.com/).

Se trata de un compendio bastante amplio de herramientas, frameworks, utilidades y servicios que se encuentran disponibles en Internet y están orientados a la recopilación de información. Se caracteriza por estar bien organizado y cada una de las secciones que componen el proyecto representa una categoría global para un tipo de información concreto.

Aunque no todos los servicios disponibles en el OSINT Framework cuentan con una API o librería para aprovechar sus funcionalidades, muchos otros, precisamente los servicios más potentes, sí que cuentan con interfaces que permiten la integración de utilidades externas y scripts desarrollados por cualquier usuario. En prácticamente todos los casos es necesario contar con una cuenta en dichos servicios, sin embargo, la mayoría de ellos se encuentran disponibles de forma gratuita. A continuación, se explican algunos de estos servicios y cómo utilizarlos desde Python.

2.1 Librerías para consultas DNS y WHOIS

Probablemente el mejor punto de partida es precisamente la consulta de registros DNS y WHOIS para un objetivo concreto. En este sentido existen varias alternativas que se pueden considerar, la primera de ellas consiste en ejecutar herramientas comunes en sistemas Linux tales como DIG, NSLOOKUP o el comando WHOIS y posteriormente, obtener y convertir los resultados de dichas herramientas en estructuras que se puedan manejar desde un script en Python, sin embargo, esto requiere un análisis manual del texto devuelto tras la ejecución del comando.

Otra alternativa menos costosa y sencilla de implementar consiste en utilizar las librerías que se encuentran disponibles en fuentes como PyPI o en repositorios públicos en GitHub.

2.1.1 Librería DNSPython

Con esta librería es posible realizar consultas a servidores DNS del mismo modo que es posible hacerlo con herramientas como dig, fierce o nslookup. Los resultados que devuelven las funciones disponibles en la herramienta se encuentran directamente formateados en estructuras de datos típicas del lenguaje, tales como listas, tuplas o diccionarios. Para instalar esta librería, se puede utilizar PIP o hacerlo directamente desde código fuente utilizando setup-tools.

Se recomienda el segundo método por dos motivos, el primero de ellos para partir de una versión actualizada de la librería y el segundo porque en el momento de redactar este documento, aunque la instalación con PIP funciona correctamente, no es posible importar el módulo "resolvers" debido a un error interno en la versión disponible en PyPI. Dicho esto, el procedimiento de instalación sería el siguiente:

```
git clone https://github.com/rthalley/dnspython

cd dnspython/

python3 setup.py install
```

La instalación debe ejecutarse con privilegios de root.

Una vez instalada la librería, se pueden ejecutar peticiones DNS directamente desde Python con los elementos disponibles en la librería. Se recomienda ver el script **1-dns.py** para ver los módulos y clases disponibles en la librería.

2.1.2 Librería pythonwhois

Aunque para los más neófitos Internet puede parecer una red carente de control y completamente descentralizada, la realidad es que existen organizaciones que se encargan de gestionar la interoperabilidad de los sistemas que funcionan en la red e intentan prevenir los conflictos que puedan producirse con las direcciones IP, así como también la gestión de los nombres de dominios, parámetros de protocolos como DNS y números de puertos.

Este tipo de funciones son llevadas a cabo desde hace varios años por una organización sin ánimo de lucro compuesta por empresas, universidades y comunidades de usuarios llamada ICANN (Internet Corporation for Assigned Names and Numbers). Aunque antiguamente estos detalles técnicos eran gestionados completamente por el gobierno de los Estados Unidos de América con la IANA (Internet Assigned Numbers Authority) en la actualidad dichas actividades son supervisadas y coordinadas por la ICANN.

IANA es ahora un departamento complementario de la ICANN. Aunque los servidores centrales de la ICANN realizan actividades de gestión sobre dominios, direcciones IP, servidores, entre otras cosas, la información necesaria para resolver direcciones y dominios se encuentra diseminada por Internet en múltiples servidores WHOIS. Dicho esto, se puede entender que WHOIS es un protocolo que permite realizar consultas con el fin de obtener información detallada sobre el propietario de un dominio, fechas de creación y caducidad, teléfonos de contacto e incluso direcciones y códigos postales.

Desde luego es una muy buena fuente de información para un atacante y un punto de inicio adecuado para perfilar un objetivo. Realizar consultas whois es un procedimiento bastante trivial en sistemas basados en Unix ejecutando utilidades por la línea de comandos, sin embargo, existen una gran variedad de servicios en Internet que permiten consultar los registros whois para un dominio determinado, por ejemplo: http://whois.net/

No obstante, resulta conveniente tener toda esta información en estructuras de datos que puedan ser utilizadas posteriormente desde un script. Es aquí donde es posible utilizar la librería **pythonwhois** que permite ejecutar consultas whois contra un dominio concreto y posteriormente almacenar los resultados en una estructura de datos consistente y fácil de utilizar. Esta librería puede ser descargada desde PyPI en el siguiente enlace: https://pypi.python.org/pypi/pythonwhois

El procedimiento de instalación de esta librería, como muchas otras disponibles en Python, consiste simplemente en ejecutar el script **setup.py** con el argumento install o utilizando el comando PIP3.
En el script **1-whois.py** se puede ver el uso de los componentes disponibles en la librería y sus funciones más interesantes.

2.2 Integración Python y Shodan

Shodan (http://www.shodan.io) es conocido como "el Google de los hackers" debido a que se trata de un potente motor de búsquedas que permite encontrar servidores y dispositivos en Internet que ejecutan servicios muy concretos.

A diferencia de los buscadores convencionales, Shodan se encarga de indexar en su base de datos interna, las cabeceras y banners correspondientes a servidores que se encuentran en ejecución en Internet. Del mismo modo que ocurre con buscadores como Google, Shodan cuenta con una serie de filtros que permiten restringir los resultados de las consultas y algunos de los más útiles se listan en la siguiente tabla.

Filtro	**Descripción**
city	En los resultados de la búsqueda, solamente aparecerán aquellos que corresponden con la ciudad indicada.
country	En los resultados de la búsqueda, solamente aparecerán aquellos que corresponden con el país indicado
geo	Permite encontrar los dispositivos que se encuentran en el radio definido por la latitud y longitud especificada.
hostname	En los resultados de la búsqueda, solamente aparecerán aquellos que corresponden con el nombre de dominio indicado.
net	Permite realizar búsquedas dirigidas a segmentos de red concretos.
os	Permite filtrar los resultados con un sistema operativo determinado.
port	Permite filtrar los resultados con un puerto determinado.

Se trata de una lista no exhaustiva y en la medida de que Shodan va creciendo, el número de filtros puede ser mayor, con lo cual, se anima al lector a leer la documentación oficial donde se incluyen los filtros soportados en https://help.shodan.io.

Por otro lado, una característica que convierte a Shodan en una herramienta imprescindible para un atacante o pentester, es que cuenta con una API que permite que desarrolladores en lenguajes de programación como Ruby o Python, puedan utilizar Shodan de forma programática, algo que desde luego resulta sumamente útil. Para utilizar la API, es necesario tener una cuenta de usuario valida y de esta forma obtener una **Developer Key**.

El uso de las características básicas de Shodan no supone ningún coste para un desarrollador, sin embargo, existen add-ons que incluyen características muy interesantes, como por ejemplo la capacidad de realizar búsquedas específicas para servicios como Telnet o HTTPS, acceso a todos los filtros disponibles desde la API y acceso sin restricciones a los resultados de las consultas. Se trata de características que no son nada despreciables y que pueden ser adquiridas por un costo muy bajo.

Para descargar e instalar la librería de Shodan para Python se sigue el mismo patrón que muchas de las librerías disponibles para este lenguaje. Es posible hacerlo utilizando easy_install, pip o directamente ejecutando el script setup.py con el argumento install.

El uso más básico de la API de Shodan para Python consiste en crear una instancia de la clase **shodan.Shodan** especificando como único argumento una **Developer Key** válida que se encuentra asociada a una cuenta de usuario. En el script **1-shodanAccount.py** se puede apreciar el uso básico de Shodan, en donde se obtiene un objeto del tipo **shodan.Shodan** y con él se realiza una búsqueda abierta con una palabra introducida por parte del usuario. Finalmente, el script enseña el número total de coincidencias con el filtro indicado y enseña únicamente los 10 primeros registros en la terminal.

2.3 SpiderFoot

Se trata de uno de los frameworks más completos en cuanto a técnicas OSINT se refiere.

Se encuentra desarrollado en Python y soporta más de 300 integraciones con servicios online para la recolección de información en fuentes abiertas. Es un proyecto opensource y cuenta con un largo recorrido y una comunidad que se ha ido afianzando en todos los años que lleva de desarrollo. Además, es un proyecto muy activo, algo que se puede apreciar en el repositorio GitHub oficial que se encuentra disponible en la siguiente URL: https://github.com/smicallef/spiderfoot

Para poder usar esta herramienta no hace falta programar absolutamente nada, aunque cuenta con un sistema de módulos que permiten extender sus funcionalidades por medio de rutinas que se pueden integrar en el framework. La documentación oficial explica el procedimiento básico para la creación de estos elementos y se puede leer en el siguiente enlace: https://www.spiderfoot.net/documentation/#modules

Para utilizar la herramienta, lo más habitual es clonar el repositorio de GitHub y a continuación, instalar las dependencias descritas en el fichero "requirements.txt" utilizando PIP. El script principal de SpiderFoot es **sf.py** y a la hora de ejecutarlo y empezar a trabajar con la herramienta, es necesario indicar la opción "-l" con la interfaz de red y puerto en el que se va a levantar un servidor web. Por ejemplo: **python3 sf.py -l 0.0.0.0:5555**

Una vez ejecutado el comando anterior, solamente basta acceder a la herramienta por medio de un navegador web

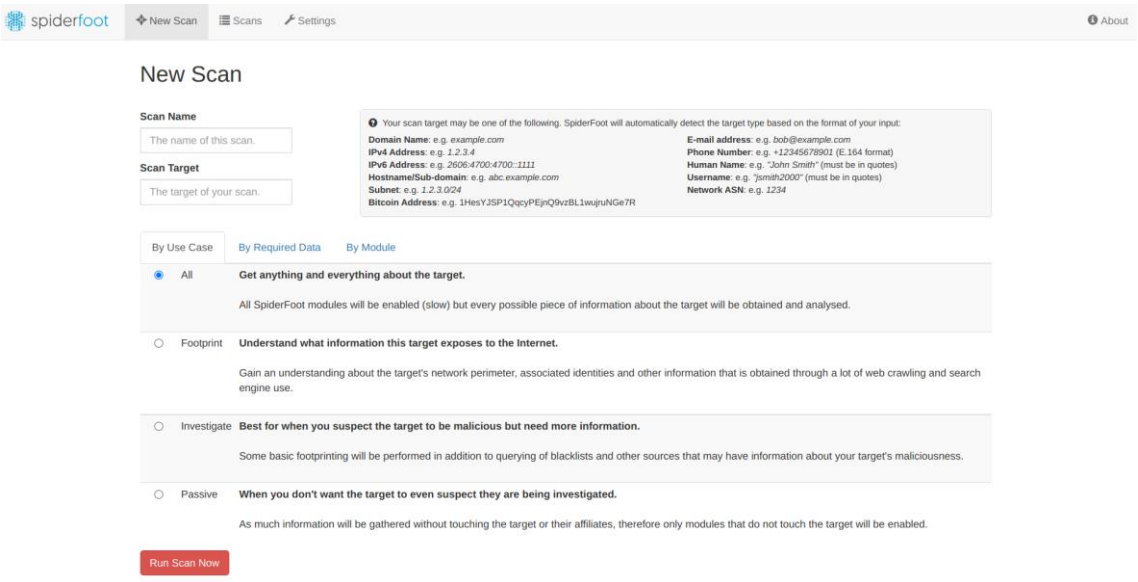

Figura 2.1: Opciones de escaneo en SpiderFoot.

El uso de la herramienta se basa simplemente en la configuración de los módulos que se encuentran disponibles en la sección de “settings”, como se puede apreciar en dicha página, aparecen múltiples integraciones con servicios de uso común para la aplicación de técnicas OSINT.

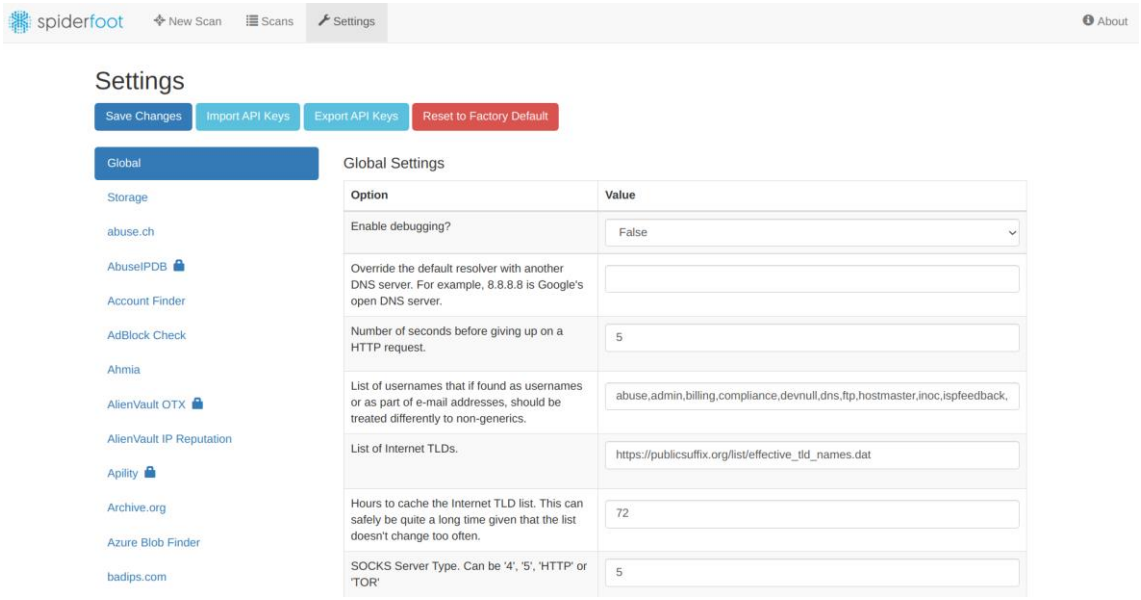

Figura 2.2: Integraciones en SpiderFoot.

Como se puede comprobar al acceder por las diferentes opciones de configuración, en la mayoría de los casos son muy simples y prácticamente no requieren ningún cambio, sin embargo, algunos servicios, como es el caso de AbuseIPDB, Censys o Shodan, requieren una API Key vinculado a una cuenta en ese servicio para su correcto funcionamiento.

Después de aplicar todas las configuraciones deseadas, es el momento de ejecutar un escaneo, algo que es tan simple como introducir en la sección “New scan” el objetivo que se desea analizar, el cual puede ser una URL, dirección IP, dominio, teléfono, etc.

Finalmente, antes de iniciar el escaneo se debe seleccionar el caso de uso concreto y ejecutar. Ahora será necesario esperar a que la herramienta proceda con la recolección de información en las fuentes que se hayan configurado previamente.

Recolección de información activa

3. Enumeración y Escaneos.

No es suficiente con recolectar información de forma pasiva, durante una campaña de Red Team o una auditoría de pentesting es necesario interactuar con el objetivo y recolectar información sobre los servicios que se encuentran en ejecución, así como las características del sistema que se pretende analizar.

En este punto es importante detectar las versiones correspondientes de los servicios en ejecución, características de configuración y si es posible, la versión exacta del sistema operativo que se está usando en dicho entorno. Esta información es útil en etapas posteriores, en las que es necesario verificar si existen vulnerabilidades conocidas sobre los servicios encontrados o una configuración insegura que le permita al atacante tener cierto nivel de control sobre el objetivo.

Para conseguir estos propósitos existen muchas herramientas disponibles actualmente, las cuales se han ido afinando y mejorando de forma continua a lo largo de los años. Un buen ejemplo de esto es Nmap, la herramienta de referencia en cuestiones de enumeración y escaneo de puertos. Es una herramienta que, aunque cuenta con muchas opciones disponibles, no es difícil de utilizar y devuelve buenos resultados, sin embargo, es importante tener conocimientos sólidos sobre redes de datos y el funcionamiento de los protocolos basados en TCP y UDP. A continuación, se describen algunas alternativas para utilizar Python a la hora de lanzar procedimientos de enumeración y reconocimiento.

3.1 Descubrimiento de sistemas en una red local.

Es posible utilizar Python desde un sistema conectado a una red local para descubrir de forma automática las máquinas que se encuentran conectadas.

El procedimiento inicial es simple, basta con ejecutar de forma iterativa el comando PING contra cada una de las máquinas que se pueden encontrar en un rango de direcciones.

Es un procedimiento simple que ayuda a recuperar rápidamente las máquinas que son capaces de atender a las peticiones ICMP. No obstante, pueden existir máquinas en el entorno que ignoren dichas peticiones, como por ejemplo sistemas Windows con el firewall activo. En esos casos la máquina no podrá ser descubierta utilizando este método, pero representa una buena aproximación inicial para entender qué máquinas se encuentran disponibles.

El script **1-ping.py** permite ejecutar el descubrimiento inicial descrito anteriormente y para ello, utiliza únicamente las clases disponibles en el lenguaje, concretamente los elementos disponibles en el módulo "subprocess". Este script se encarga de escribir en un fichero de texto cada una de las IPs descubiertas, lo que facilita posteriormente su análisis. Funciona tanto en sistemas Windows como en sistemas basados en Unix ya que realiza las comprobaciones oportunas con el fin de ejecutar el comando "ping" de forma correcta.

3.2 Enumeración y escaneos online.

Aunque como se ha mencionado anteriormente, utilizar Nmap resulta una de las alternativas más convenientes y extendidas, sin embargo, en la etapa de reconocimiento puede ser interesante ejecutar un escaneo inicial de puertos utilizando la infraestructura de un tercero. En este sentido existen varios servicios online que permiten ejecutar un escaneo de puertos contra un dominio o dirección IP en Internet sin exponer la IP del usuario que ha solicitado el escaneo. En este sentido, la librería scanless (https://github.com/vesche/scanless) es una de las más interesantes ya que permite ejecutar escaneos desde cualquier script en Python utilizando los servicios online más habituales para este tipo de actividades. En el script **1-scanless.py** se puede apreciar el uso de esta librería y sus beneficios. Tal como se indica en la documentación aportada en el repositorio de GitHub, el proceso de instalación puede realizarse utilizando PIP y se puede especificar cualquiera de los servicios de escaneo disponibles.

3.3 Integración entre Python y Nmap.

Dado que Nmap es una de las herramientas más comunes en el arsenal de un pentester, no es de extrañar que existan varias librerías en Python que permitan llevar a cabo una integración entre dicha herramienta y Python para automatizar su uso. En PyPI existen varias alternativas que funcionan correctamente, sin embargo, en este caso se utilizará "python-nmap" (https://pypi.org/project/python-nmap/) dada la flexibilidad que ofrece a la hora de ejecutar escaneos personalizados. La librería permite ejecutar el proceso de escaneo con Nmap de forma asíncrona, esto quiere decir que se puede ejecutar el escaneo en segundo plano y continuar normalmente con la ejecución del script. En tal caso, cada vez que ocurra un evento definido; como por ejemplo encontrar un puerto abierto, se invocará una función de "callback" predefinida para gestionar dicho evento.

Por otro lado, el uso básico de la librería consiste en utilizar la clase PortScanner que ejecuta un escaneo con Nmap de forma síncrona, esto significa que el script quedará en estado de espera hasta que el proceso de escaneo finalice su ejecución.
Para instalar python-nmap es necesario tener en cuenta que hay dos versiones distintas, una que funciona para las versiones de Python 2.7 o anteriores y otra que está diseñada específicamente para las versiones de Python 3.x. En ambos casos, es necesario descargar el paquete desde el sitio web oficial ubicado en el siguiente enlace:

http://xael.org/norman/python/python-nmap/

Y posteriormente, ejecutar el script **setup.py** con el argumento install o bien utilizar PIP para Python 3.x o 2.7 según sea el caso.

El script **1-nmap.py** utiliza esta librería, pero además, cuenta con una característica muy interesante y es la posibilidad de parsear la estructura JSON devuelta por la librería con los resultados y posteriormente, volcar dicha información en objetos Python. El objetivo de esto es precisamente manejar de una forma más cómoda los resultados del escaneo.

4 PUNTOS CLAVE

- La recolección de información pasiva representa el primer paso en cualquier proceso de pentesting y se caracteriza precisamente por la ausencia de interacción directa con el objetivo.
- El uso de técnicas OSINT representa el primer paso a la hora de entender el funcionamiento del sistema objetivo y las relacionados que puede tener con otros sistemas o componentes externos.
- En Python existen varias librerías y componentes para ejecutar técnicas OSINT, sin embargo, uno de los proyectos que mejores resultados aporta es Shodan, el cual cuenta con una librería concreta para Python
- Nmap es una de las herramientas más importantes en el mundo del pentesting y permite recolectar información de forma activa.
- Existen múltiples librerías disponibles en Python para ejecución de escaneos con Nmap, sin embargo, todas ellas tienen en común que devuelven objetos o estructuras de datos con la información devuelta por la herramienta.